Pattern Search for Mixed Variable Optimization Problems

Mark A. Abramson (Air Force Institute of Technology)

Charles Audet (École Polytechnique de Montréal)

John Dennis, Jr. (Rice University)

The views expressed here are those of the authors and do not reflect the official policy or position of the USAF, Department of Defense, US government, or our other sponsors:

Boeing, ExxonMobil, CSRI, LASCI, NSF.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **27 FEB 2003**	2. REPORT TYPE	3. DATES COVERED **00-00-2003 to 00-00-2003**
4. TITLE AND SUBTITLE **Pattern Search for Mixed Variable Optimization Problems**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Air Force Institute of Technology,Wright Patterson AFB,OH,45433**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**96**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Outline

- Mixed variable problem motivation and formulation

Outline

- Mixed variable problem motivation and formulation
- GPS for linearly constrained MVP problems

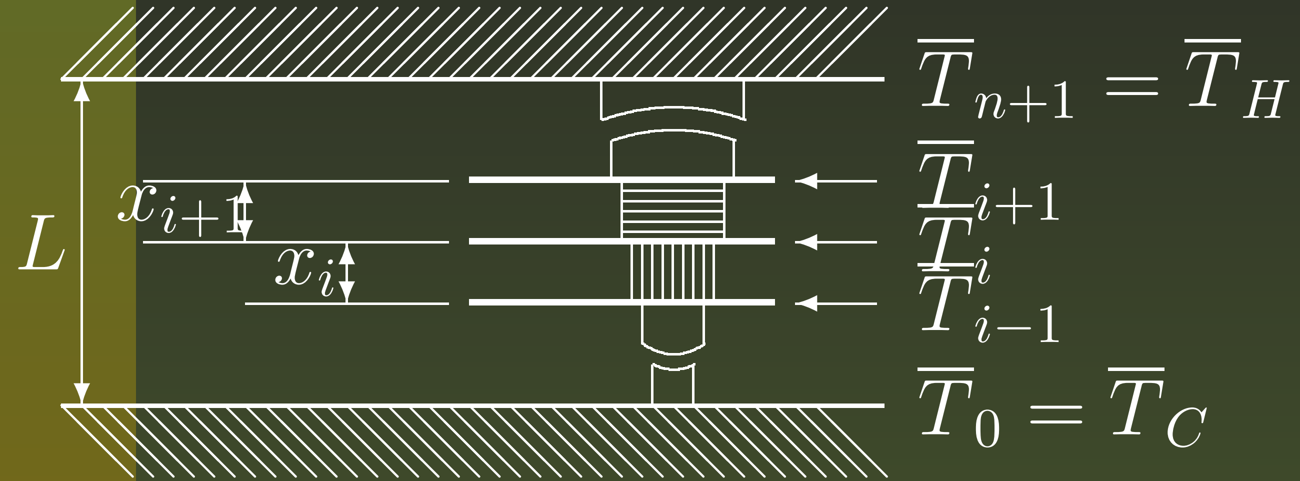
Outline

- Mixed variable problem motivation and formulation
- GPS for linearly constrained MVP problems
- Filter GPS for general constrained MVP problems

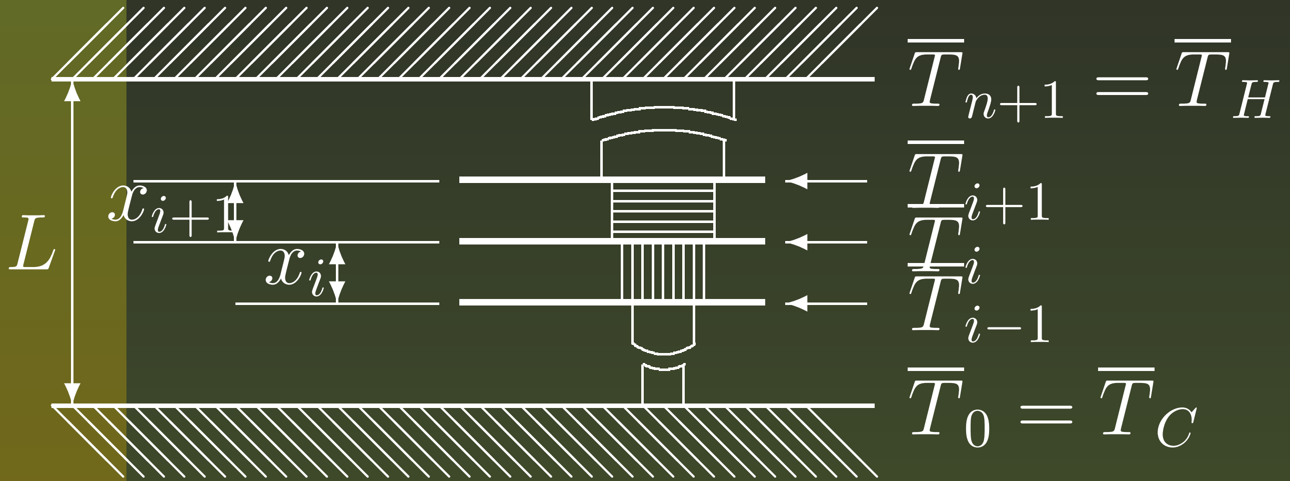
Outline

- Mixed variable problem motivation and formulation
- GPS for linearly constrained MVP problems
- Filter GPS for general constrained MVP problems
- Results for thermal insulation system design

Heat intercept insulation system

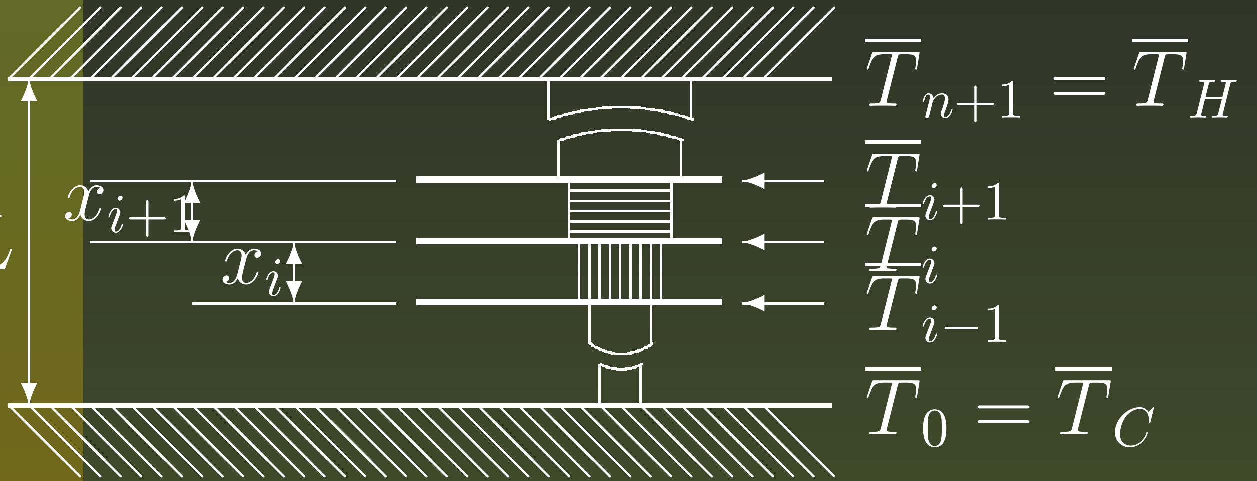


Heat intercept insulation system



$$\min \quad power(n, I, x, \overline{T})$$

Heat intercept insulation system



$$\begin{aligned} \min \quad & power(n, I, x, \overline{T}) \\ \text{subject to} \quad & n \in \{1, 2, \ldots, n_{\max}\}, \quad I \in \mathcal{I}^{n+1} \\ & \overline{T}_{i-1} \leq \overline{T}_i \leq \overline{T}_{i+1},\ i = 1, 2, \ldots, n \\ & \sum_{i=1}^{n+1} x_i = L, \quad x_i \geq 0,\ i = 1, 2, \ldots, n+1 \end{aligned}$$

The general MVP problem

$$\min_{x \in X} f(x)$$

$$\text{subject to } C(x) \leq 0,$$

The general MVP problem

$$\min_{x \in X} f(x)$$

$$\text{subject to } C(x) \leq 0,$$

where

- $x = (x^c, x^d) \in \Re^{n^c} \times \mathcal{Z}^{n^d}$
- $X = X^c \times X^d$, where
  $X^c(x^d) = \{x^c \in \Re^{n^c} : \ell(x^d) \leq A(x^d)x^c \leq u(x^d)\}$

The general MVP problem

$$\min_{x \in X} f(x)$$
$$\text{subject to } C(x) \leq 0,$$

where

- $x = (x^c, x^d) \in \Re^{n^c} \times \mathcal{Z}^{n^d}$
- $X = X^c \times X^d$, where $X^c(x^d) = \{x^c \in \Re^{n^c} : \ell(x^d) \leq A(x^d)x^c \leq u(x^d)\}$
- f and $C = (c_1, c_2, \ldots, c_p)$ may be discontinuous, extended valued, costly

The general MVP problem

$$\min_{x \in X} f(x)$$
$$\text{subject to } C(x) \leq 0,$$

where

- $x = (x^c, x^d) \in \Re^{n^c} \times \mathcal{Z}^{n^d}$
- $X = X^c \times X^d$, where
  $X^c(x^d) = \{x^c \in \Re^{n^c} : \ell(x^d) \leq A(x^d)x^c \leq u(x^d)\}$
- f and $C = (c_1, c_2, \ldots, c_p)$ may be discontinuous, extended valued, costly

Choices other than pattern search

Some methods that come to mind are:

- MINLP methods: cannot handle categorical variables

Choices other than pattern search

Some methods that come to mind are:

- MINLP methods: cannot handle categorical variables
- Search heuristics: huge numbers of evaluations and very limited convergence theory

Choices other than pattern search

Some methods that come to mind are:

- MINLP methods: cannot handle categorical variables
- Search heuristics: huge numbers of evaluations and very limited convergence theory
  - Simulated annealing

Choices other than pattern search

Some methods that come to mind are:

- MINLP methods: cannot handle categorical variables
- Search heuristics: huge numbers of evaluations and very limited convergence theory
  - Simulated annealing
  - Tabu search

Choices other than pattern search

Some methods that come to mind are:

- **MINLP methods**: cannot handle categorical variables
- **Search heuristics**: huge numbers of evaluations and very limited convergence theory
  - Simulated annealing
  - Tabu search
  - Evolutionary algorithms

Choices other than pattern search

Some methods that come to mind are:

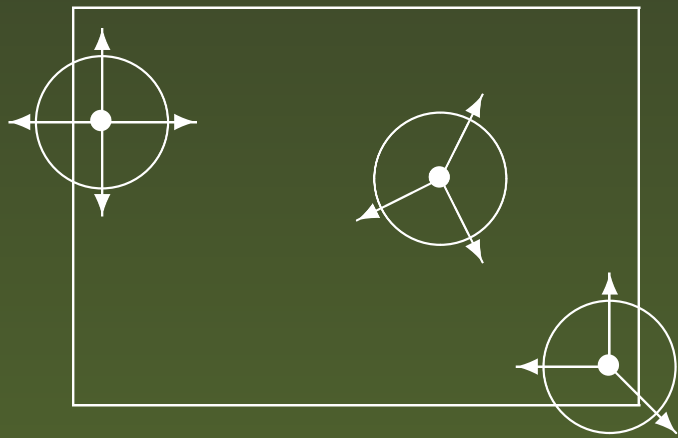
- **MINLP methods**: cannot handle categorical variables
- **Search heuristics**: huge numbers of evaluations and very limited convergence theory
  - Simulated annealing
  - Tabu search
  - Evolutionary algorithms
- **Other methods**: SQP/direct search with 1 categorical variable

Generalized pattern searches

INITIALIZATION of directions and step size

For $k = 1, 2, \ldots$

- SEARCH a finite set of mesh points



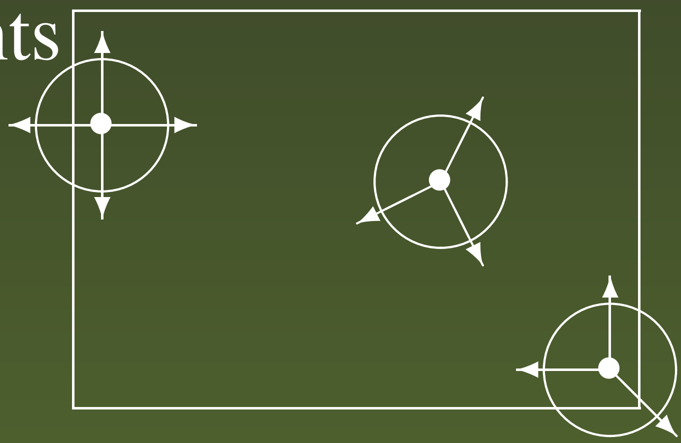
End

Generalized pattern searches

INITIALIZATION of directions and step size

For $k = 1, 2, \ldots$

- SEARCH a finite set of mesh points
- POLL neighboring mesh points



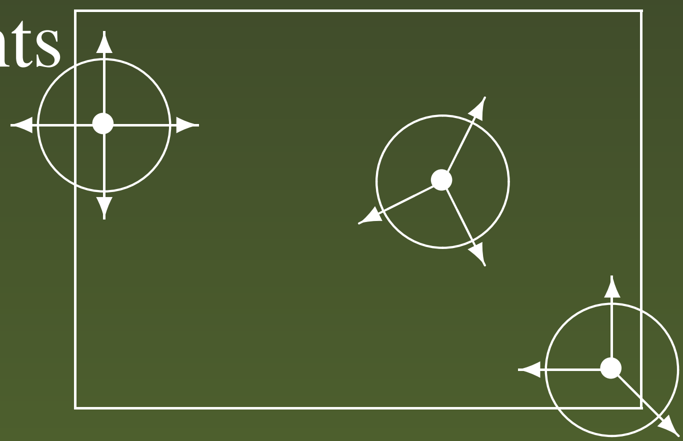
End

Generalized pattern searches

INITIALIZATION of directions and step size

For $k = 1, 2, \ldots$

- SEARCH a finite set of mesh points
- POLL neighboring mesh points
- UPDATE parameters:



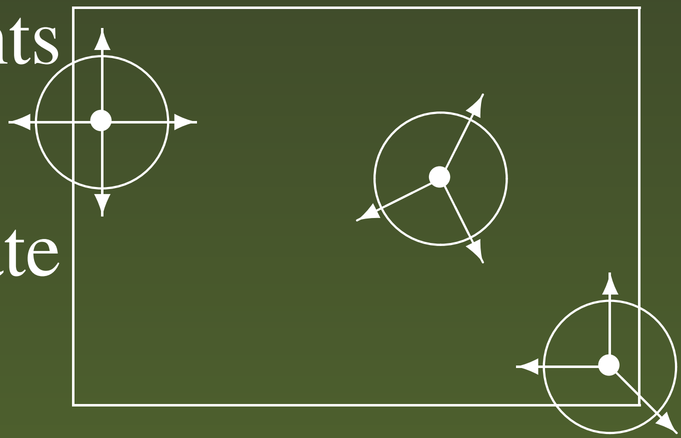
End

Generalized pattern searches

INITIALIZATION of directions and step size

For $k = 1, 2, \ldots$

- SEARCH a finite set of mesh points
- POLL neighboring mesh points
- UPDATE parameters:
  - Success: Accept new iterate



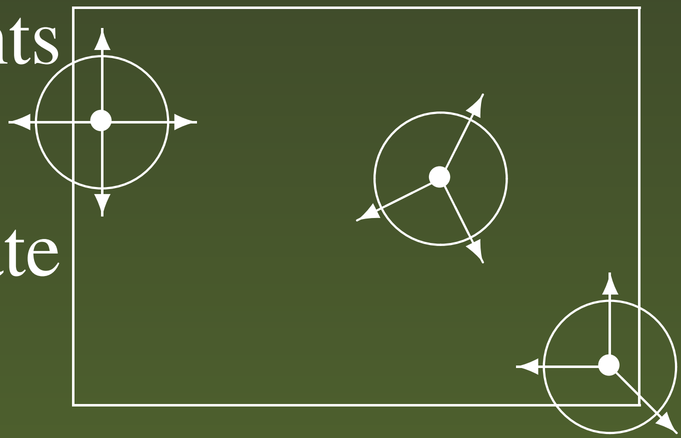
End

Generalized pattern searches

INITIALIZATION of directions and step size

For $k = 1, 2, \ldots$

- SEARCH a finite set of mesh points
- POLL neighboring mesh points
- UPDATE parameters:
  - Success: Accept new iterate
  - Failure: Refine mesh



End

Details of kth POLL step

Mesh: $M_k = \{p_k + \Delta_k D z : z \in \mathcal{Z}_+^{|D|}\}$,
Poll set: $P_k = \{p_k + \Delta_k d : d \in D_k \subseteq D\}$,

Details of kth POLL step

Mesh: $M_k = \{p_k + \Delta_k Dz : z \in \mathcal{Z}_+^{|D|}\}$,
Poll set: $P_k = \{p_k + \Delta_k d : d \in D_k \subseteq D\}$,

where

- p_k is the current poll center
- $\Delta_k > 0$ is the mesh size parameter
- D_k, D are positive spanning sets

Details of kth POLL step

Mesh: $M_k = \{p_k + \Delta_k Dz : z \in \mathcal{Z}_+^{|D|}\}$,
Poll set: $P_k = \{p_k + \Delta_k d : d \in D_k \subseteq D\}$,

where

- p_k is the current poll center
- $\Delta_k > 0$ is the mesh size parameter
- D_k, D are positive spanning sets

Examples: $D = [I, -I] \quad D = [I, -\mathbf{e}]$

One of these directions should be a descent direction.

Derivative information can reduce poll set to a singleton

Details of kth POLL step

Mesh: $M_k = \{p_k + \Delta_k Dz : z \in \mathcal{Z}_+^{|D|}\}$,
Poll set: $P_k = \{p_k + \Delta_k d : d \in D_k \subseteq D\}$,

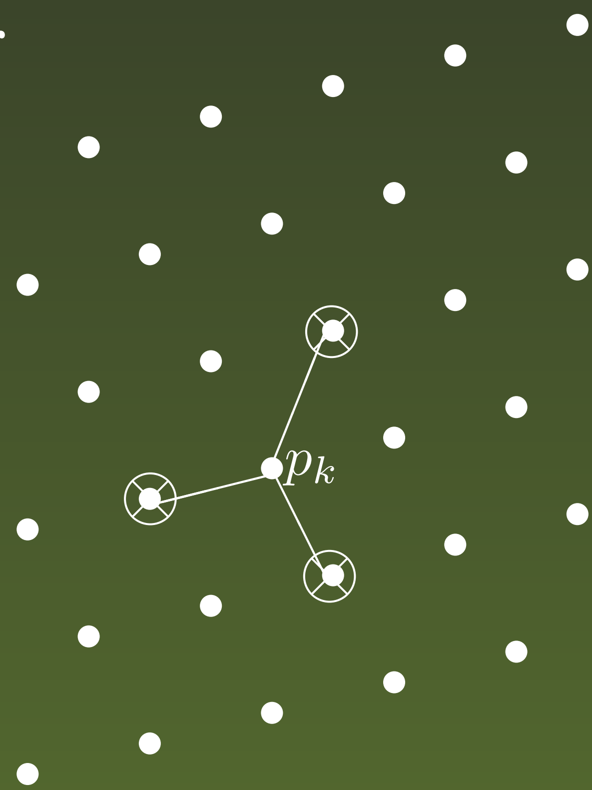
where

- p_k is the current poll center
- $\Delta_k > 0$ is the mesh size parameter
- D_k, D are positive spanning sets

Examples: $D = [I, -I] \quad D = [I, -\mathbf{e}]$

One of these directions should be a descent direction.

Derivative information can reduce poll set to a singleton



Definition of local optimality

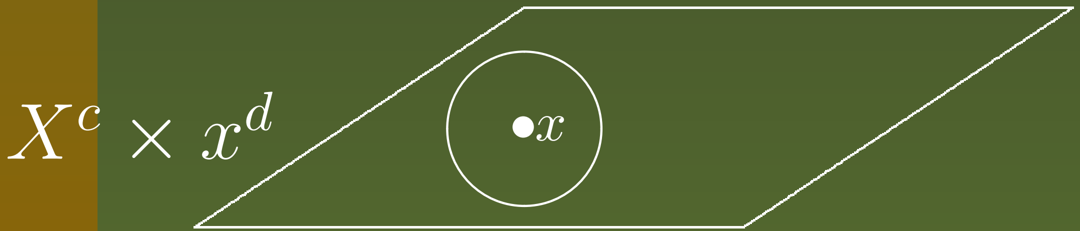
$x = (x^c, x^d) \in X$ is a *local minimizer of* f *with respect to neighbors* $\mathcal{N}(x) \subset X$ if $\exists \epsilon > 0$ such that $f(x) \leq f(v)$

$$\forall v \in X \cap \bigcup_{y \in \mathcal{N}(x)} \left(B(y^c, \epsilon) \times y^d \right).$$

Definition of local optimality

$x = (x^c, x^d) \in X$ is a *local minimizer of* f *with respect to neighbors* $\mathcal{N}(x) \subset X$ if $\exists \epsilon > 0$ such that $f(x) \le f(v)$

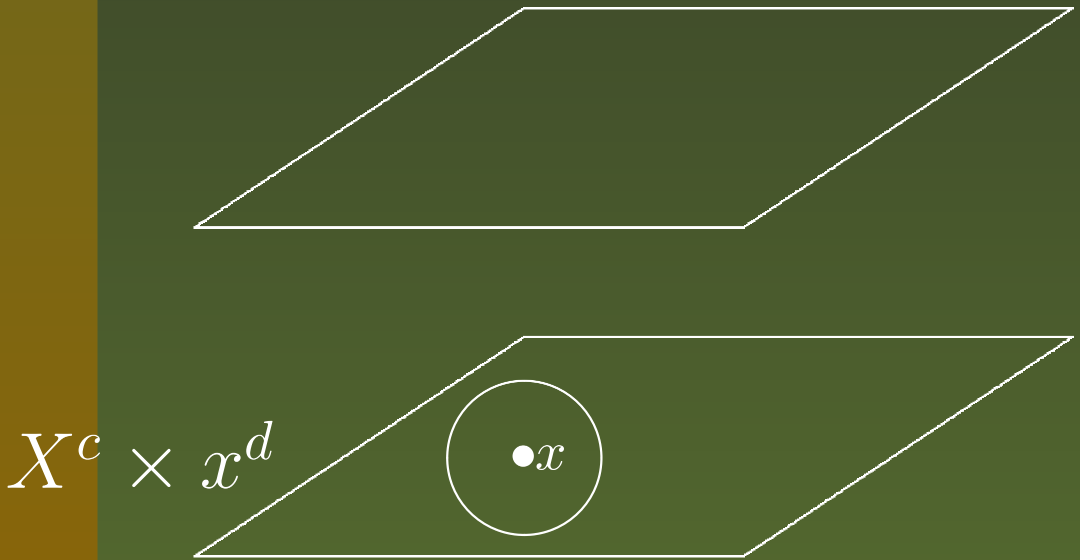
$$\forall v \in X \cap \bigcup_{y \in \mathcal{N}(x)} \left(B(y^c, \epsilon) \times y^d \right).$$



Definition of local optimality

$x = (x^c, x^d) \in X$ is a *local minimizer of f with respect to neighbors* $\mathcal{N}(x) \subset X$ if $\exists \epsilon > 0$ such that $f(x) \leq f(v)$

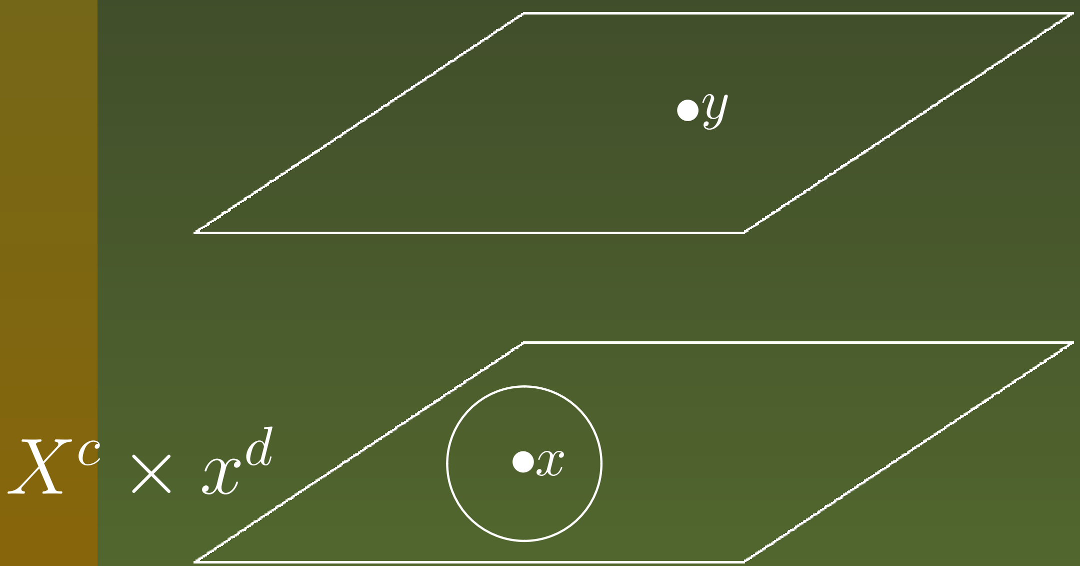
$$\forall v \in X \cap \bigcup_{y \in \mathcal{N}(x)} \left(B(y^c, \epsilon) \times y^d \right).$$



Definition of local optimality

$x = (x^c, x^d) \in X$ is a *local minimizer of f with respect to neighbors* $\mathcal{N}(x) \subset X$ if $\exists \epsilon > 0$ such that $f(x) \leq f(v)$

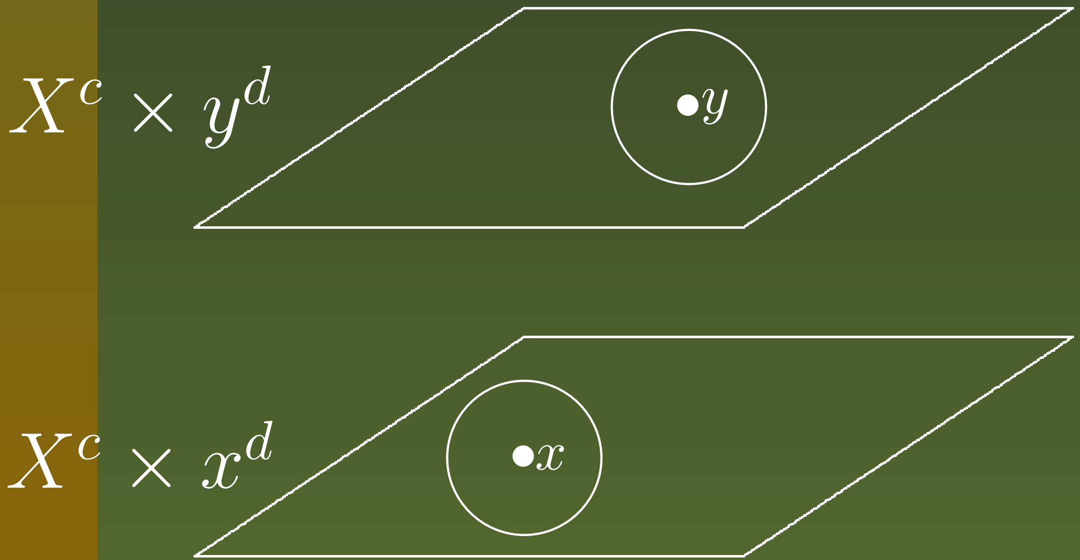
$$\forall v \in X \cap \bigcup_{y \in \mathcal{N}(x)} \left(B(y^c, \epsilon) \times y^d \right).$$



Definition of local optimality

$x = (x^c, x^d) \in X$ is a *local minimizer of* f *with respect to neighbors* $\mathcal{N}(x) \subset X$ if $\exists \epsilon > 0$ such that $f(x) \leq f(v)$

$$\forall v \in X \cap \bigcup_{y \in \mathcal{N}(x)} \left(B(y^c, \epsilon) \times y^d \right).$$



Heatshield discrete neighbors

- Replace any single insulator with a different type

Heatshield discrete neighbors

- Replace any single insulator with a different type
- Remove any intercept with its left insulator, and increase the thickness of its right insulator to absorb the deficit

Heatshield discrete neighbors

- Replace any single insulator with a different type
- Remove any intercept with its left insulator, and increase the thickness of its right insulator to absorb the deficit
- Add an intercept at any position:

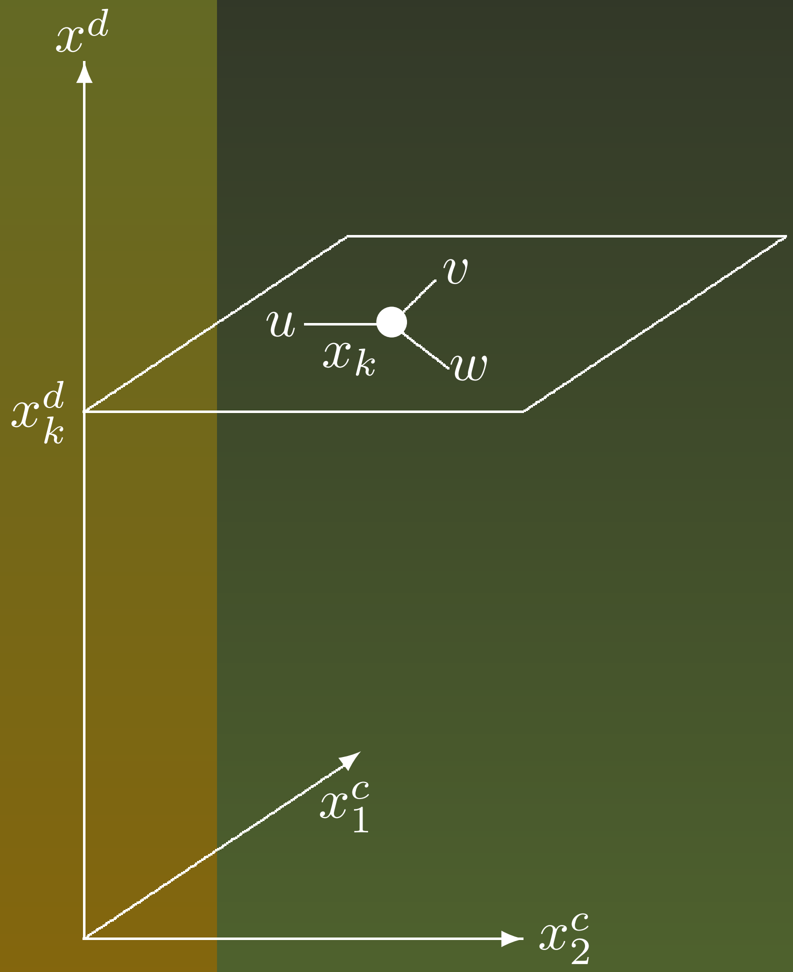
Heatshield discrete neighbors

- Replace any single insulator with a different type
- Remove any intercept with its left insulator, and increase the thickness of its right insulator to absorb the deficit
- Add an intercept at any position:
  - The existing insulator is divided (rounded to the mesh)

Heatshield discrete neighbors

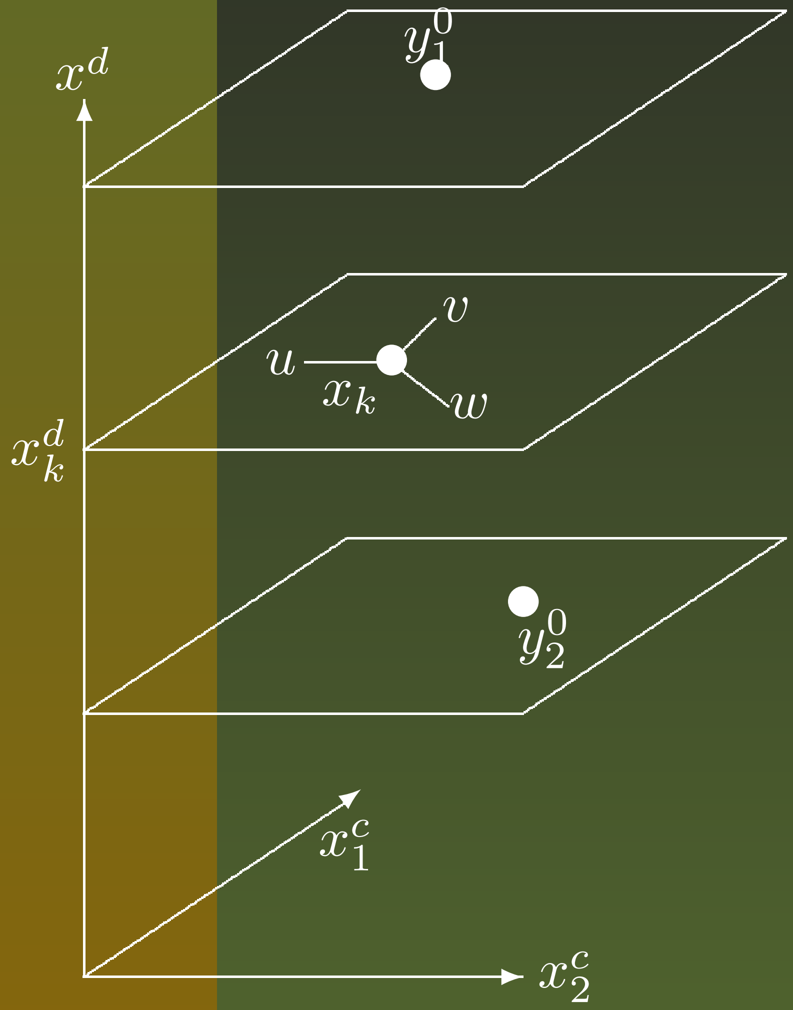
- Replace any single insulator with a different type
- Remove any intercept with its left insulator, and increase the thickness of its right insulator to absorb the deficit
- Add an intercept at any position:
  - The existing insulator is divided (rounded to the mesh)
  - The cooling temperature is set to the average of the two intercepts adjacent to it (rounded to the mesh)

Construction of the poll set



$$P_k = \{u, v, w\}$$

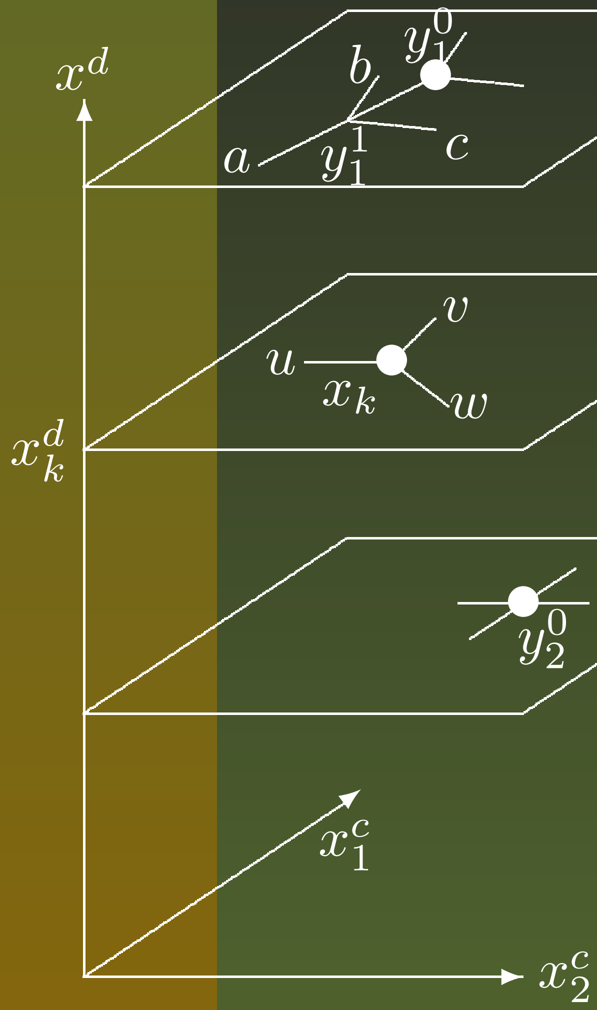
Construction of the poll set



$$P_k = \{u, v, w\}$$
$$\mathcal{N}(x_k) = \{x_k, y_1^0, y_2^0\}$$

$y_1^0 \in \mathcal{N}(x_k)$ satisfies
$$f(x_k) < f(y_1^0) < f(x_k) + \xi$$

Construction of the poll set

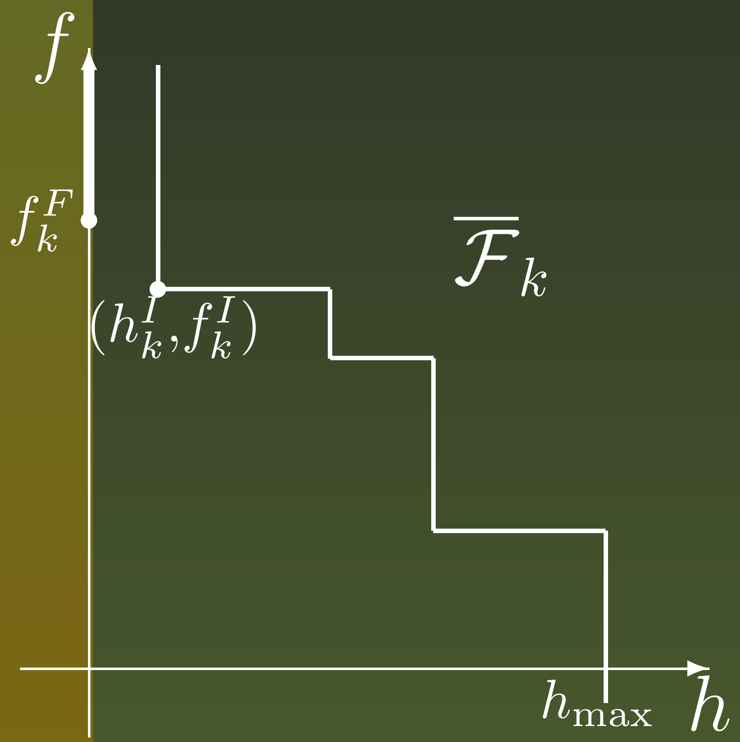


$$P_k = \{u, v, w\}$$
$$\mathcal{N}(x_k) = \{x_k, y_1^0, y_2^0\}$$
$$\mathcal{X}_k = \{y_1^1\} \cup \{a, b, c\}$$

$y_1^0 \in \mathcal{N}(x_k)$ satisfies
$$f(x_k) < f(y_1^0) < f(x_k) + \xi$$

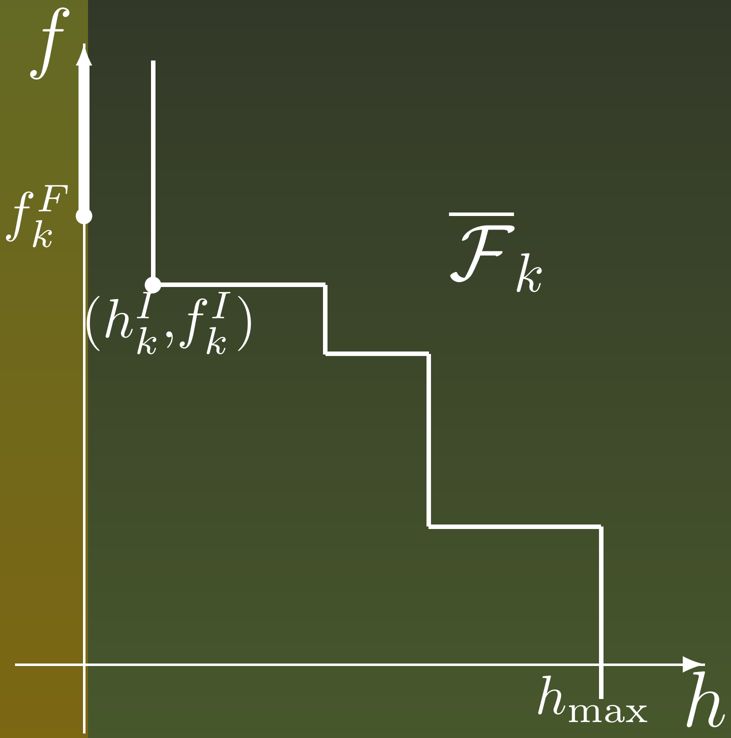
Poll Set: $P_k \cup \mathcal{N}(x_k) \cup \mathcal{X}_k$

Filter GPS for nonlinear constraints



$$h(x) = \|C(x)_+\|^2$$

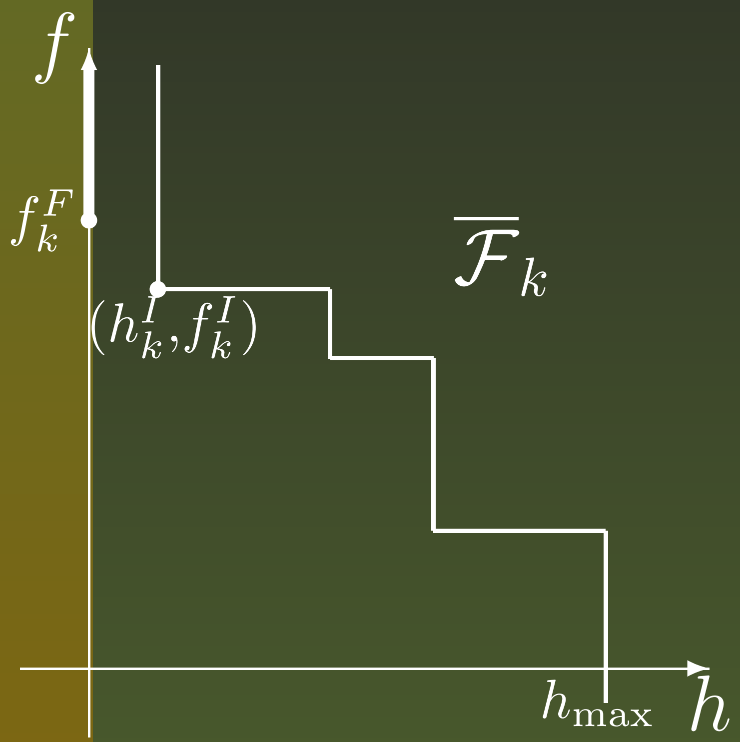
Filter GPS for nonlinear constraints



Poll center is either best feasible point or least infeasible point.

$$h(x) = \|C(x)_+\|^2$$

Filter GPS for nonlinear constraints

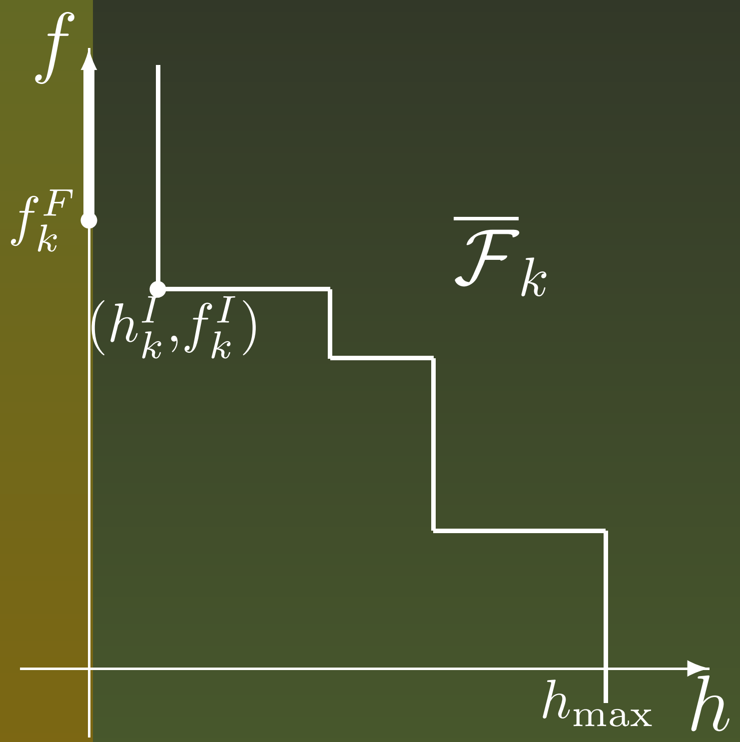


$$h(x) = \|C(x)_+\|^2$$

Poll center is either best feasible point or least infeasible point.

For each trial point x, $h(x)$ and $f(x)$ are plotted on the bi-loss map.

Filter GPS for nonlinear constraints



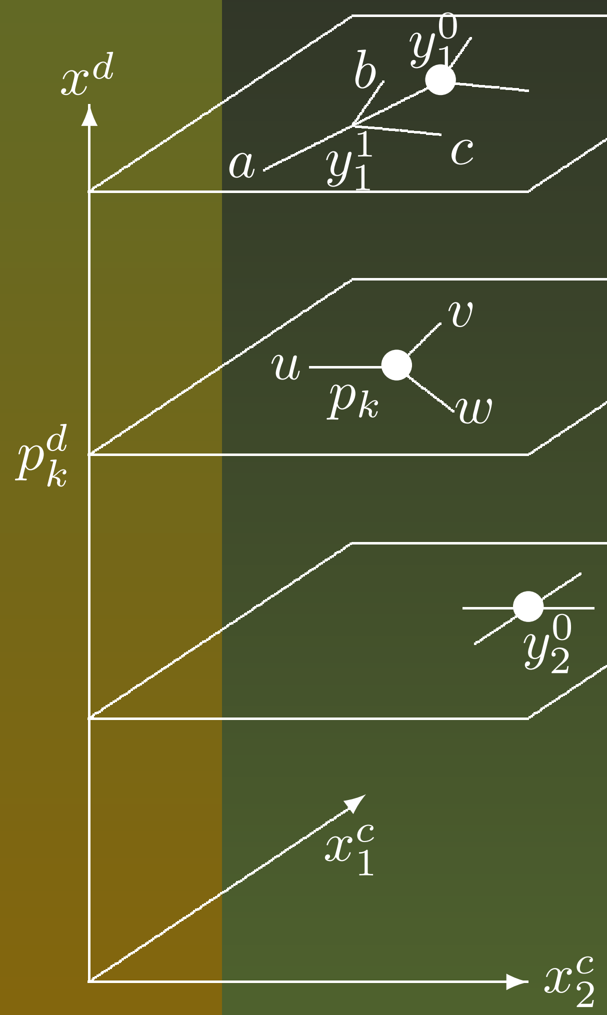
$$h(x) = \|C(x)_+\|^2$$

Poll center is either best feasible point or least infeasible point.

For each trial point x, $h(x)$ and $f(x)$ are plotted on the bi-loss map.

If x is unfiltered, it is added to the filter; otherwise, the mesh is refined.

Construction of the poll set

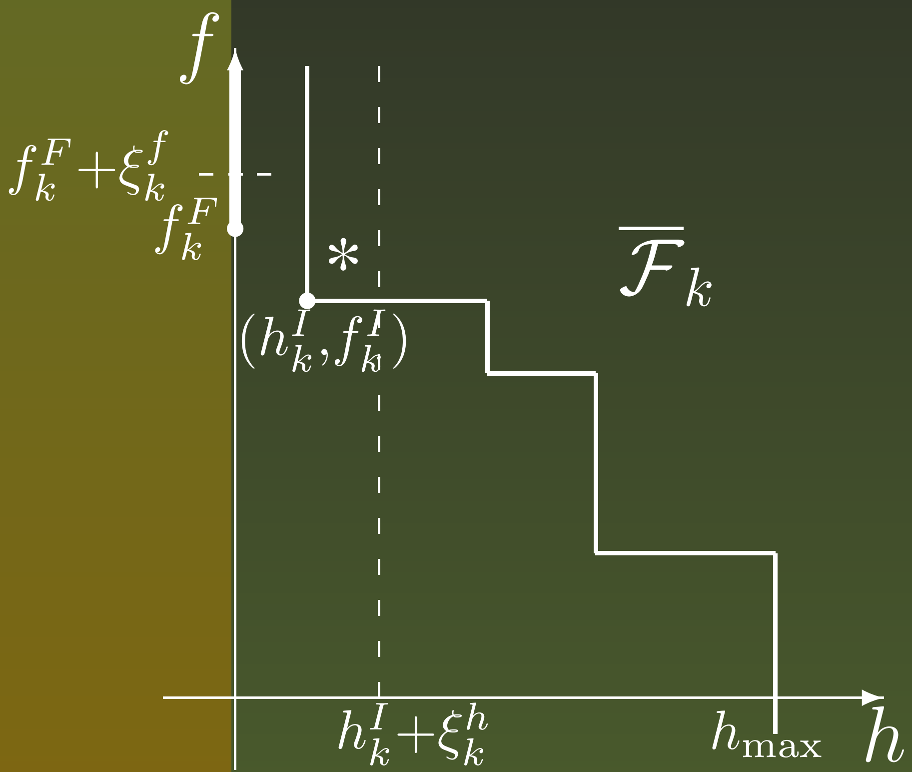


$$P_k = \{u, v, w\}$$
$$\mathcal{N}(p_k) = \{p_k, y_1^0, y_2^0\}$$
$$\mathcal{X}_k = \{y_1^1\} \cup \{a, b, c\}$$

$y_1^0 \in \mathcal{N}(p_k)$ satisfies the extended poll criteria

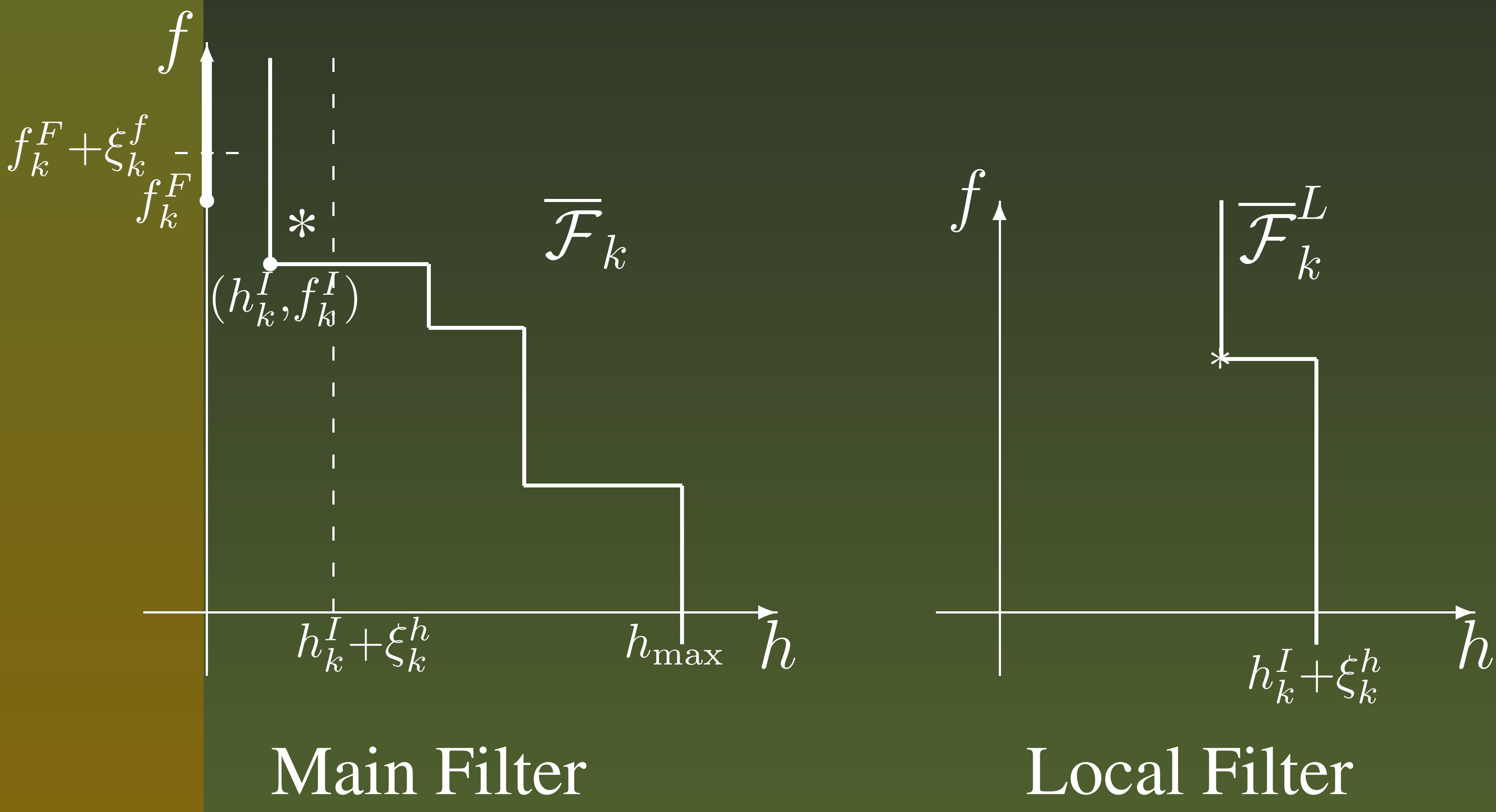
Poll Set: $P_k \cup \mathcal{N}(p_k) \cup \mathcal{X}_k$

Local filter for extended polling



Main Filter

Local filter for extended polling



Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:
  - SEARCH: Any finite strategy on the mesh

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:
  - SEARCH: Any finite strategy on the mesh
  - POLL: Evaluate points in $P_k \cup \mathcal{N}(p_k)$

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:
  - SEARCH: Any finite strategy on the mesh
  - POLL: Evaluate points in $P_k \cup \mathcal{N}(p_k)$
  - EXTENDED POLL: Evaluate points in $\mathcal{X}_k(\xi_k^f, \xi_k^h)$

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:
  - SEARCH: Any finite strategy on the mesh
  - POLL: Evaluate points in $P_k \cup \mathcal{N}(p_k)$
  - EXTENDED POLL: Evaluate points in $\mathcal{X}_k(\xi_k^f, \xi_k^h)$
- Update:

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:
  - SEARCH: Any finite strategy on the mesh
  - POLL: Evaluate points in $P_k \cup \mathcal{N}(p_k)$
  - EXTENDED POLL: Evaluate points in $\mathcal{X}_k(\xi_k^f, \xi_k^h)$
- Update:
  - If (found), set $\Delta_{k+1} \geq \Delta_k$ and update filter

Filter GPS algorithm for MVP

INITIALIZATION: Set $\Delta_0, \xi > 0$, and populate filter
For $k = 1, 2, \ldots,$ do

- Update poll center p_k and extended poll triggers ξ_k^f, ξ_k^h
- Compute incumbent values f_k^F, f_k^I, h_k^I
- While trial points are filtered do:
  - SEARCH: Any finite strategy on the mesh
  - POLL: Evaluate points in $P_k \cup \mathcal{N}(p_k)$
  - EXTENDED POLL: Evaluate points in $\mathcal{X}_k(\xi_k^f, \xi_k^h)$
- Update:
  - If (found), set $\Delta_{k+1} \geq \Delta_k$ and update filter
  - If (not found), set $\Delta_{k+1} < \Delta_k$

Convergence theory assumptions

- All iterates lie in a compact set

Convergence theory assumptions

- All iterates lie in a compact set
- The linear constraint matrix A is rational

Convergence theory assumptions

- All iterates lie in a compact set
- The linear constraint matrix A is rational
- The mesh directions conform to the geometry of X^c

Convergence theory assumptions

- All iterates lie in a compact set
- The linear constraint matrix A is rational
- The mesh directions conform to the geometry of X^c
- All discrete neighbors lie on the current mesh

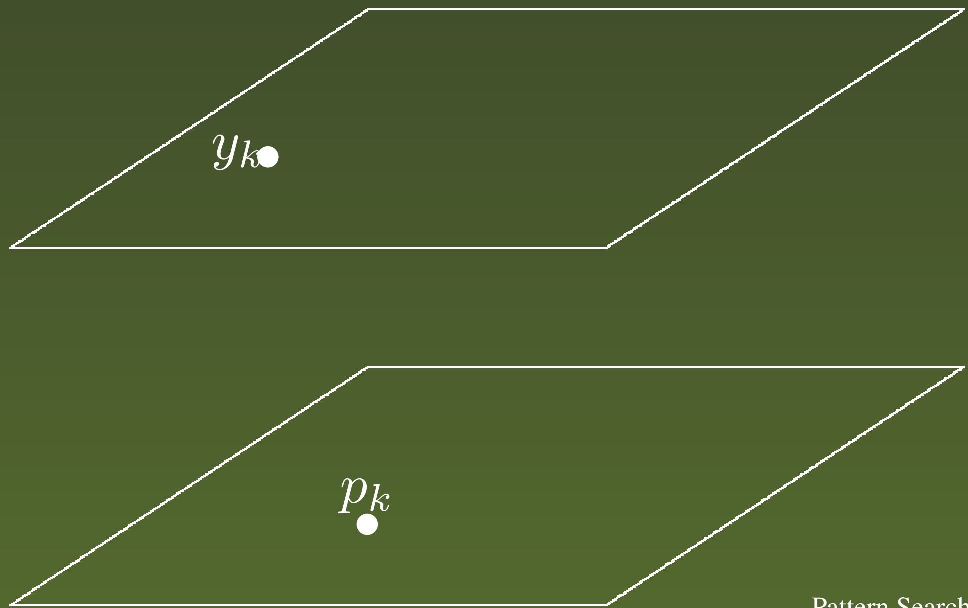
Convergence theory assumptions

- All iterates lie in a compact set
- The linear constraint matrix A is rational
- The mesh directions conform to the geometry of X^c
- All discrete neighbors lie on the current mesh
- The set-valued neighborhood function $\mathcal{N} : X \to 2^X$ satisfies a notion of continuity.

Limit points of the algorithm

$\exists$ subsequence K such that $\lim_{k \in K} \Delta_k = 0$, with limit points:

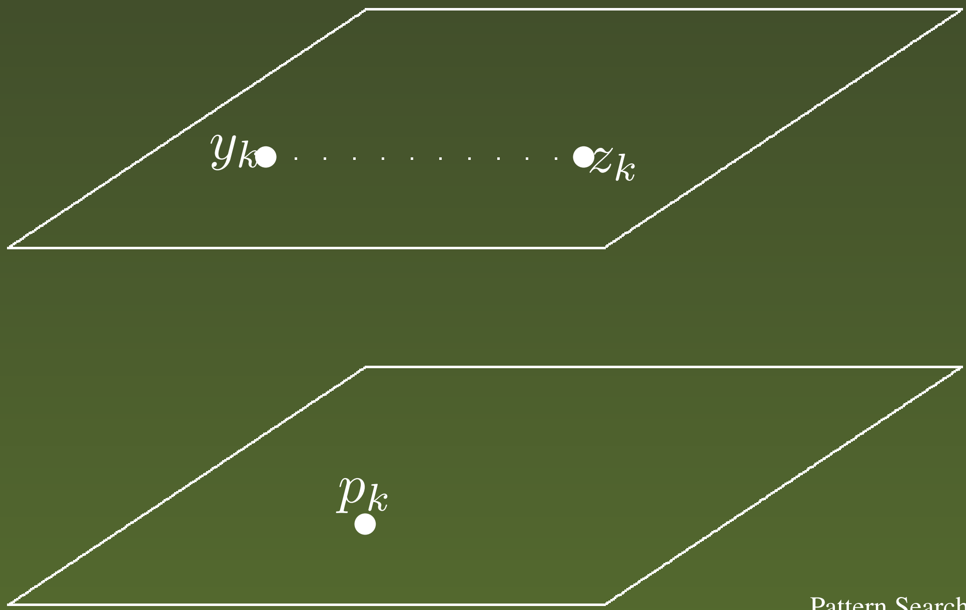
1. $\hat{p} = \lim_{k \in K} p_k$, where $p_k \in \{p_k^F, p_k^I\}$
2. $\hat{y} = \lim_{k \in K} y_k$, where $y_k \in \mathcal{N}(p_k)$ and $\hat{y} \in \mathcal{N}(\hat{p})$.
3. $\hat{z} = \lim_{k \in K} z_k$, where z_k are extended poll endpoints.



Limit points of the algorithm

$\exists$ subsequence K such that $\lim_{k \in K} \Delta_k = 0$, with limit points:

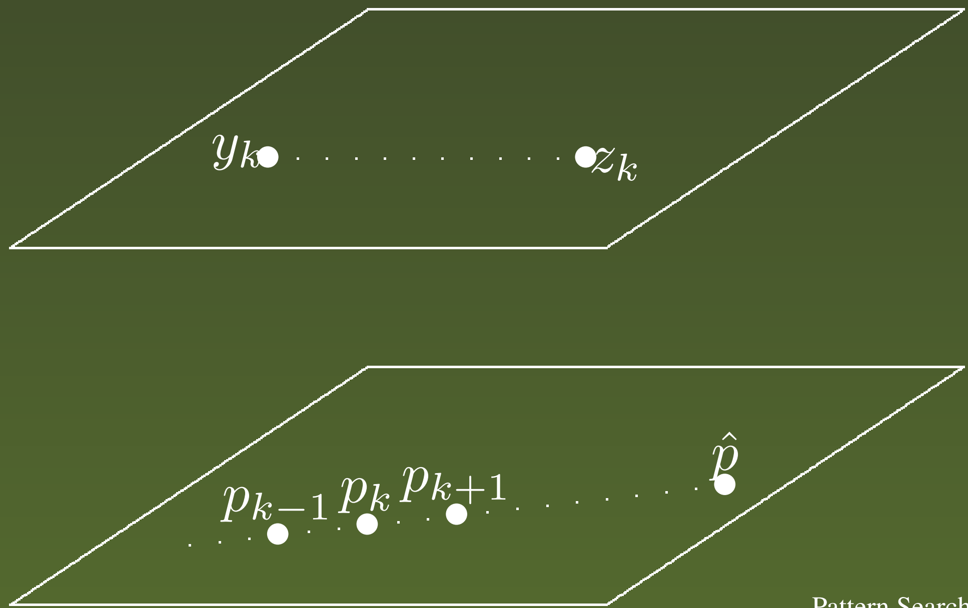
1. $\hat{p} = \lim_{k \in K} p_k$, where $p_k \in \{p_k^F, p_k^I\}$
2. $\hat{y} = \lim_{k \in K} y_k$, where $y_k \in \mathcal{N}(p_k)$ and $\hat{y} \in \mathcal{N}(\hat{p})$.
3. $\hat{z} = \lim_{k \in K} z_k$, where z_k are extended poll endpoints.



Limit points of the algorithm

$\exists$ subsequence K such that $\lim_{k \in K} \Delta_k = 0$, with limit points:

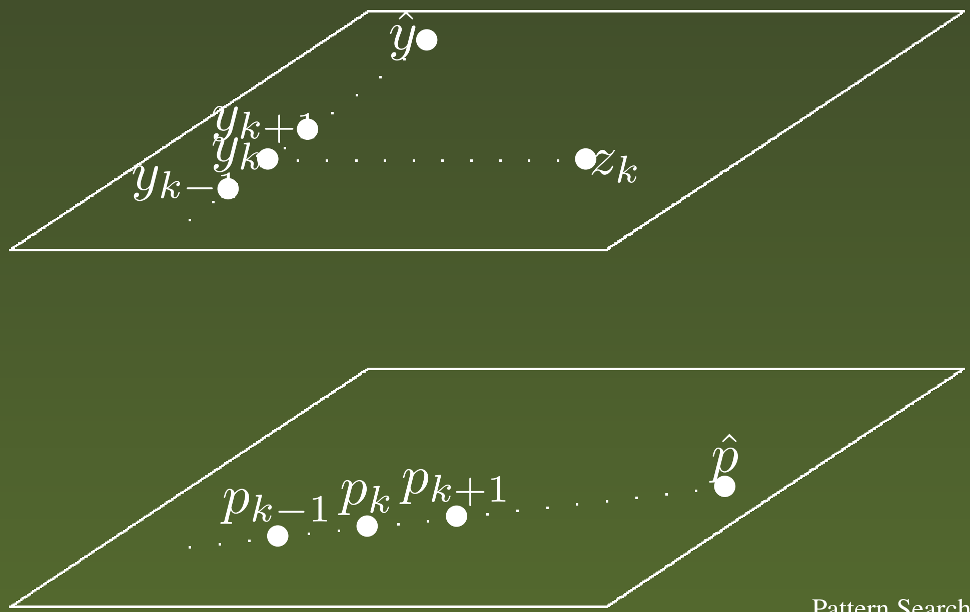
1. $\hat{p} = \lim_{k \in K} p_k$, where $p_k \in \{p_k^F, p_k^I\}$
2. $\hat{y} = \lim_{k \in K} y_k$, where $y_k \in \mathcal{N}(p_k)$ and $\hat{y} \in \mathcal{N}(\hat{p})$.
3. $\hat{z} = \lim_{k \in K} z_k$, where z_k are extended poll endpoints.



Limit points of the algorithm

$\exists$ subsequence K such that $\lim_{k \in K} \Delta_k = 0$, with limit points:

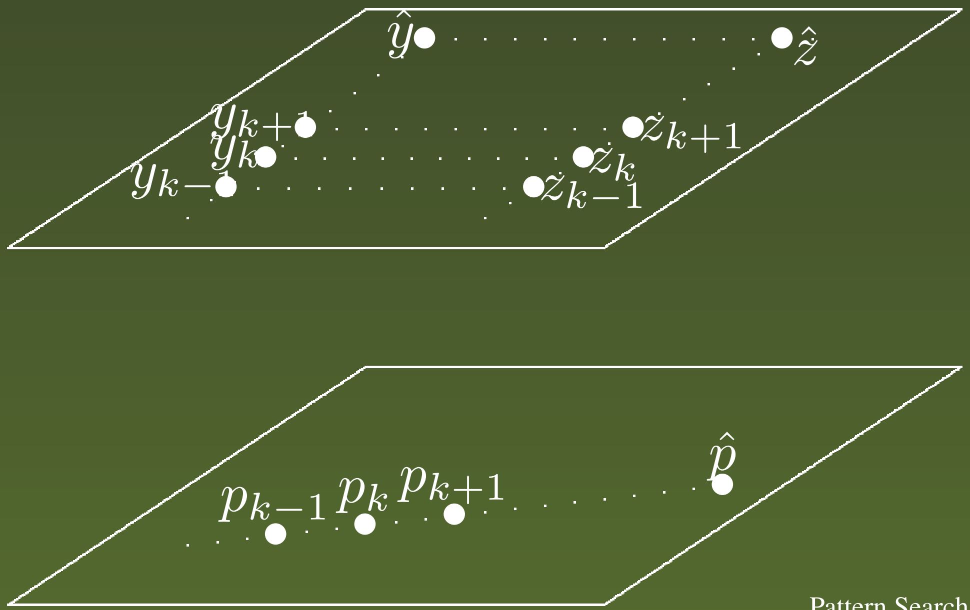
1. $\hat{p} = \lim_{k \in K} p_k$, where $p_k \in \{p_k^F, p_k^I\}$
2. $\hat{y} = \lim_{k \in K} y_k$, where $y_k \in \mathcal{N}(p_k)$ and $\hat{y} \in \mathcal{N}(\hat{p})$.
3. $\hat{z} = \lim_{k \in K} z_k$, where z_k are extended poll endpoints.



Limit points of the algorithm

$\exists$ subsequence K such that $\lim_{k \in K} \Delta_k = 0$, with limit points:

1. $\hat{p} = \lim_{k \in K} p_k$, where $p_k \in \{p_k^F, p_k^I\}$
2. $\hat{y} = \lim_{k \in K} y_k$, where $y_k \in \mathcal{N}(p_k)$ and $\hat{y} \in \mathcal{N}(\hat{p})$
3. $\hat{z} = \lim_{k \in K} z_k$, where z_k are extended poll endpoints



Filter convergence results

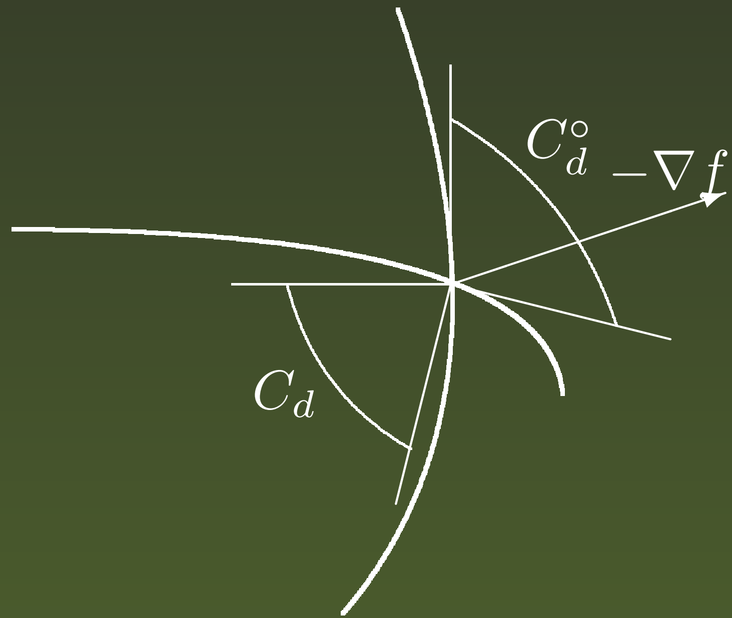
Let $D(\hat{p})$ be the set of polling directions used i.o.

- h continuous* at $\hat{p}$ and $\hat{y} \;\Rightarrow\; h(\hat{p}) \leq h(\hat{y})$
- f continuous* at $\hat{p}$ and $\hat{y}$ and $p_k = p_k^F$ i. o.
  $\Rightarrow f(\hat{p}) \leq f(\hat{y})$
- h Lipschitz* near $\hat{p} \;\Rightarrow\; h^\circ(\hat{p}; (d, 0)) \geq 0 \; \forall d \in D(\hat{p})$
- f Lipschitz* near $\hat{p}$ and $p_k = p_k^F$ i. o.
  $\Rightarrow f^\circ(\hat{p}; (d, 0)) \geq 0 \; \forall d \in D(\hat{p})$
- h strictly differentiable* at $\hat{p}$ and $\;\Rightarrow\; \nabla h(\hat{p}) = 0$

Similar results hold for certain $\hat{z}$

Filter convergence results

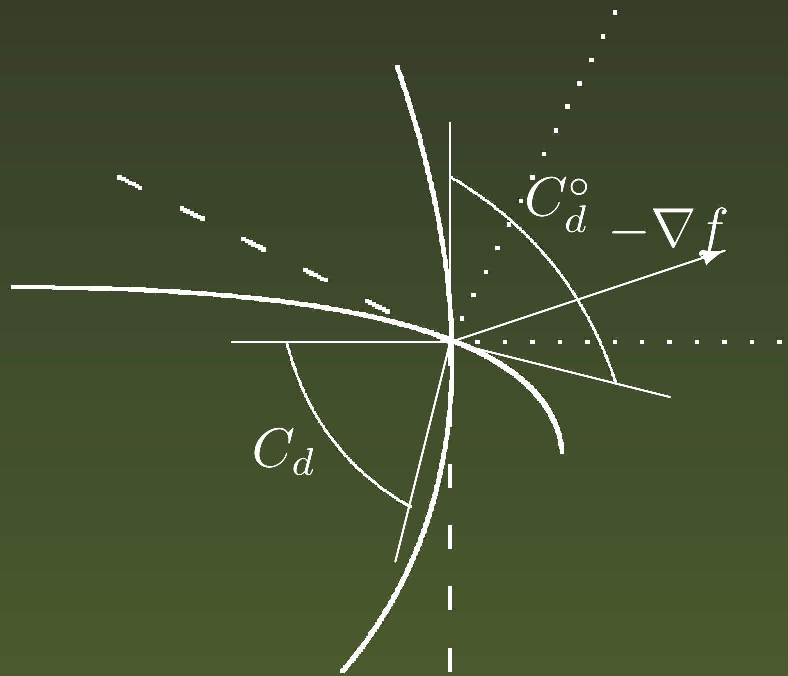
f strictly differentiable* at $\hat{p}$ and $p_k = p_k^F$ i. o.
$\Rightarrow \; -\nabla^c f(\hat{p}) \in C_d^\circ$. Similar results hold for certain $\hat{z}$



* with respect to the continuous variables

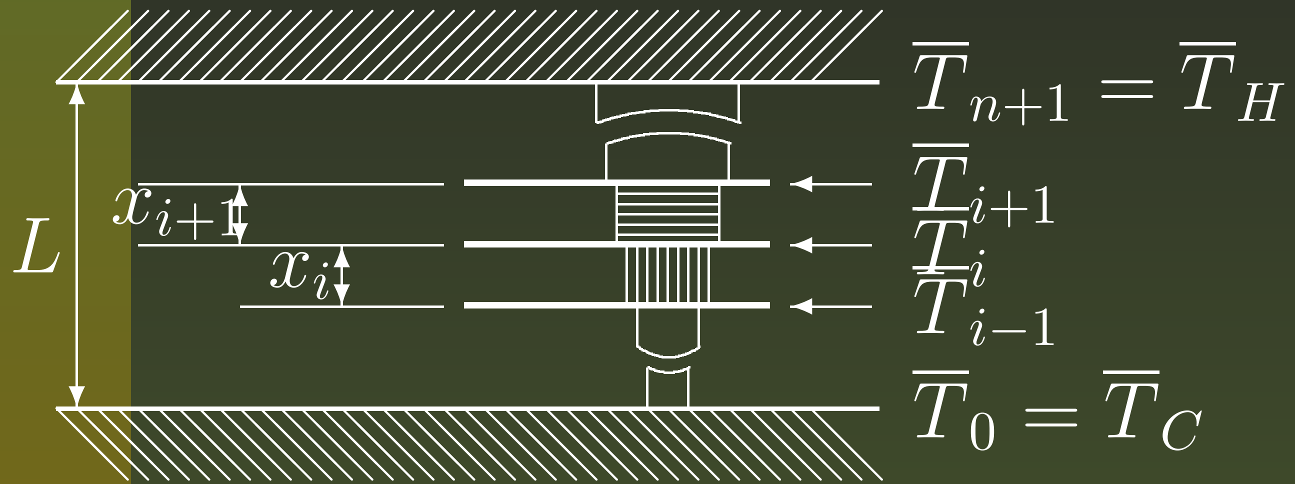
Filter convergence results

f strictly differentiable* at $\hat{p}$ and $p_k = p_k^F$ i. o.
$\Rightarrow \; -\nabla^c f(\hat{p}) \in C_d^\circ$. Similar results hold for certain $\hat{z}$

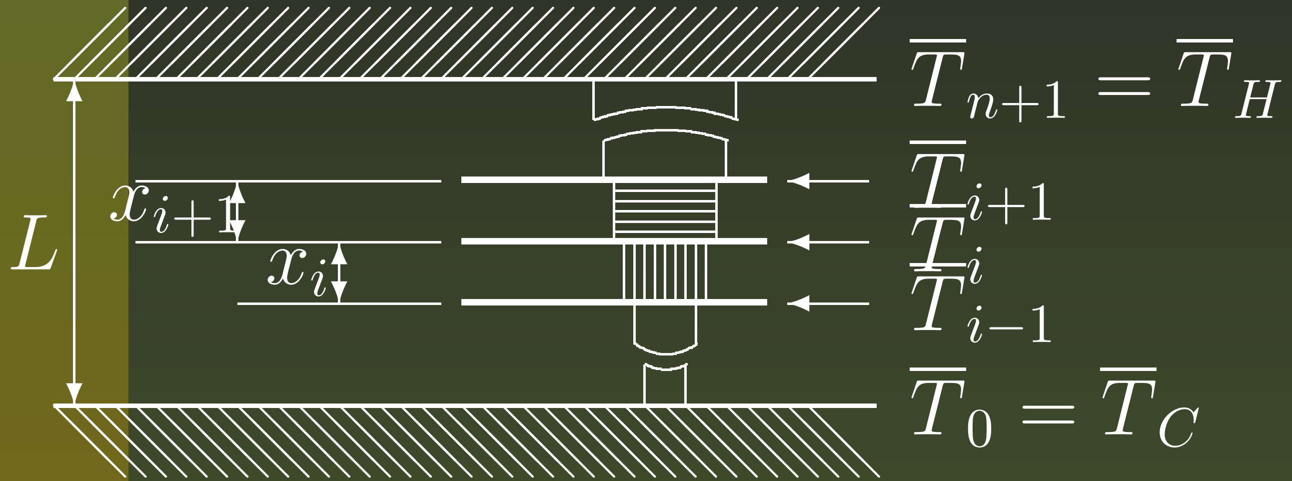


* with respect to the continuous variables

Heat intercept insulation system

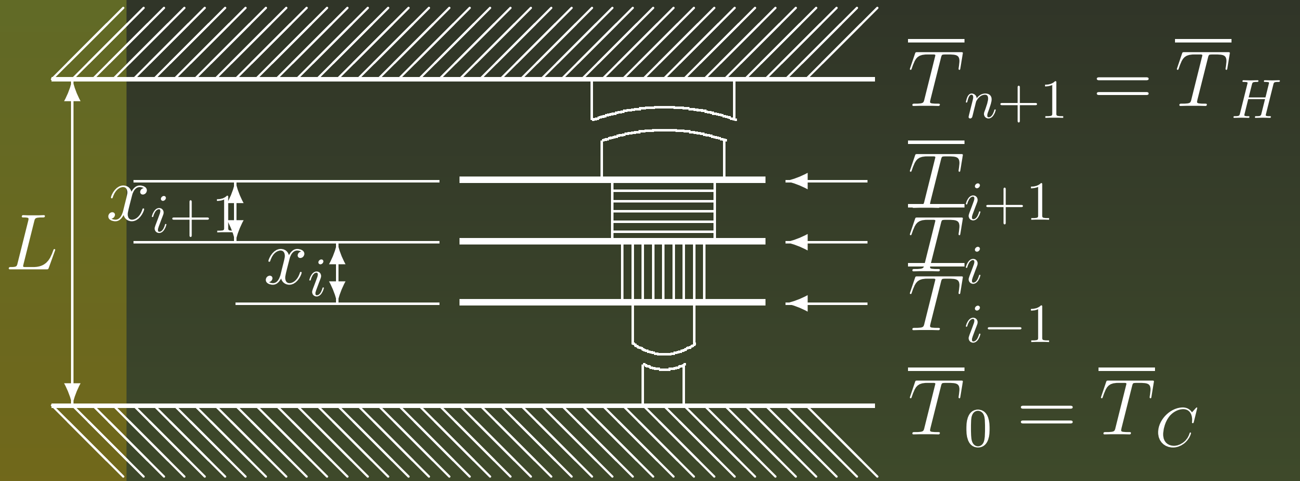


Heat intercept insulation system



$$\min \quad power(n, I, x, \overline{T})$$

Heat intercept insulation system



$$\begin{array}{rl} \min & power(n, I, x, \overline{T}) \\ \text{subject to} & n \in \{1, 2, \ldots, n_{\max}\}, \quad I \in \mathcal{I}^{n+1} \\ & \overline{T}_{i-1} \le \overline{T}_i \le \overline{T}_{i+1},\ i = 1, 2, \ldots, n \\ & \displaystyle\sum_{i=1}^{n+1} x_i = L, \quad x_i \ge 0,\ i = 1, 2, \ldots, n+1 \end{array}$$

Previous heat shield studies

- **Hilal & Boom**: 1-3 intercepts, single insulator type, constant cross-sectional areas

Previous heat shield studies

- **Hilal & Boom**: 1-3 intercepts, single insulator type, constant cross-sectional areas
- **Hilal & Eyssa**: 1-3 intercepts, single insulator type, variable cross-sectional areas

Previous heat shield studies

- **Hilal & Boom**: 1-3 intercepts, single insulator type, constant cross-sectional areas
- **Hilal & Eyssa**: 1-3 intercepts, single insulator type, variable cross-sectional areas
- **Kokkolaras, Audet & Dennis**: Variable number of intercepts, multiple insulator types, constant cross-sectional areas

Previous heat shield studies

- **Hilal & Boom**: 1-3 intercepts, single insulator type, constant cross-sectional areas
- **Hilal & Eyssa**: 1-3 intercepts, single insulator type, variable cross-sectional areas
- **Kokkolaras, Audet & Dennis**: Variable number of intercepts, multiple insulator types, constant cross-sectional areas
- **Current work**: Variable number of intercepts, multiple insulator types, variable cross-sectional areas, load-bearing nonlinear constraints

Nomenclature

$$\begin{aligned}
k(T;I_i) &= \text{Thermal conductivity function for insulator } i \\
P_i &= \text{Power applied to intercept } i \\
C_i &= \text{Thermodynamic cycle coefficient at intercept } i \\
q_i &= \text{Heat flow from intercept } i \text{ to } i-1 \\
A_i &= \text{Cross-sectional area of insulator } i \\
\sigma(T;I_i) &= \text{maximum allowable stress function} \\
e(T;I_i) &= \text{unit thermal expansion function} \\
\rho(I_i) &= \text{density of the insulator } i \text{ material} \\
F &= \text{load (force) to be placed on the system} \\
m_{\max} &= \text{maximum allowable mass of the insulators} \\
\delta &= \text{maximum allowable \% thermal contraction}
\end{aligned}$$

Heat shield objective and nonlinear constraints

- Minimize **power**: $\sum_{i=1}^{n} P_i = \sum_{i=1}^{n} C_i \left(\frac{\overline{T}_H}{\overline{T}_i} - 1 \right) (q_i - q_{i-1})$

  By Fourier's law: $q_i = \frac{A_i}{x_i} \int_{\overline{T}_{i-1}}^{\overline{T}_i} k(T; I_i) dT, \quad i = 1, 2, \ldots, n+1$

Heat shield objective and nonlinear constraints

- Minimize **power**: $\displaystyle\sum_{i=1}^{n} P_i = \sum_{i=1}^{n} C_i \left(\frac{\overline{T}_H}{\overline{T}_i} - 1 \right) (q_i - q_{i-1})$

  By Fourier's law: $\displaystyle q_i = \frac{A_i}{x_i} \int_{\overline{T}_{i-1}}^{\overline{T}_i} k(T; I_i)\, dT, \quad i = 1, 2, \ldots, n+1$

- **Stress**: $\displaystyle\frac{F}{A_i} \leq \min\{\sigma(T; I_i) : \overline{T}_{i-1} \leq T \leq \overline{T}_i\}$ (binding)

Heat shield objective and nonlinear constraints

- Minimize **power**: $\displaystyle\sum_{i=1}^{n} P_i = \sum_{i=1}^{n} C_i \left(\frac{\overline{T}_H}{\overline{T}_i} - 1\right)(q_i - q_{i-1})$

  By Fourier's law: $\displaystyle q_i = \frac{A_i}{x_i}\int_{\overline{T}_{i-1}}^{\overline{T}_i} k(T; I_i)\,dT, \quad i = 1, 2, \ldots, n+1$

- **Stress**: $\displaystyle\frac{F}{A_i} \leq \min\{\sigma(T; I_i) : \overline{T}_{i-1} \leq T \leq \overline{T}_i\}$ (binding)

- **Mass**: $\displaystyle\sum_{i=1}^{n+1} \rho(I_i) A_i x_i \leq m_{\max}$

Heat shield objective and nonlinear constraints

- Minimize **power**: $\sum_{i=1}^{n} P_i = \sum_{i=1}^{n} C_i \left(\frac{\overline{T}_H}{\overline{T}_i} - 1 \right) (q_i - q_{i-1})$

  By Fourier's law: $q_i = \frac{A_i}{x_i} \int_{\overline{T}_{i-1}}^{\overline{T}_i} k(T; I_i) dT, \quad i = 1, 2, \ldots, n+1$

- **Stress**: $\frac{F}{A_i} \leq \min\{\sigma(T; I_i) : \overline{T}_{i-1} \leq T \leq \overline{T}_i\}$ (binding)

- **Mass**: $\sum_{i=1}^{n+1} \rho(I_i) A_i x_i \leq m_{\max}$

- **Contraction**: $\sum_{i=1}^{n+1} \left(\frac{\int_{\overline{T}_{i-1}}^{\overline{T}_i} e(T; I_i) k(T; I_i) dT}{\int_{\overline{T}_{i-1}}^{\overline{T}_i} k(T; I_i) dT} \right) \left(\frac{x_i}{L} \right) \leq \frac{\delta}{100}$

Heat shield implementation

- **Materials**:

Nylon	6063-T5 Aluminum
Teflon	Fiberglass Epoxy (normal cloth)
304 Stainless Steel	Fiberglass Epoxy (plane cloth)
1020 Low Carbon Steel	

Heat shield implementation

- **Materials**:

  Nylon

  Teflon

  304 Stainless Steel

  1020 Low Carbon Steel

  6063-T5 Aluminum

  Fiberglass Epoxy (normal cloth)

  Fiberglass Epoxy (plane cloth)

- **Material Data from Lookup Tables or Graphs**:

  Thermal Conductivity

  Unit Thermal Contraction

  Maximum Allowable Stress (Tensile Yield Strength)

Heat shield implementation

- **Materials**:

Nylon	6063-T5 Aluminum
Teflon	Fiberglass Epoxy (normal cloth)
304 Stainless Steel	Fiberglass Epoxy (plane cloth)
1020 Low Carbon Steel	

- **Material Data from Lookup Tables or Graphs**:

  Thermal Conductivity Unit Thermal Contraction

  Maximum Allowable Stress (Tensile Yield Strength)

- **Interpolation/Integration**: Cubic splines, Simpson's rule

Heat shield implementation

- **Materials**:

Nylon	6063-T5 Aluminum
Teflon	Fiberglass Epoxy (normal cloth)
304 Stainless Steel	Fiberglass Epoxy (plane cloth)
1020 Low Carbon Steel	

- **Material Data from Lookup Tables or Graphs**:

  Thermal Conductivity Unit Thermal Contraction

  Maximum Allowable Stress (Tensile Yield Strength)

- **Interpolation/Integration**: Cubic splines, Simpson's rule
- **Search/Poll**: No Search, Poll around p_k^F

Heat shield computational results

Source	$\frac{PL}{A}\left[\frac{W}{cm}\right]$	Insulators
Hilal & Boom	68.6	$E_pE_pE_p$
Kokkolaras et al.	25.3	$NNNNNNNEEET$
Kokkolaras rerun	25.59	$NNNNNNTEETT$
Hilal & Eyssa	53.2	$E_pE_pE_p$
with stress constraint	24.55	$EEEEEEEEEEE$
with all constraints	23.77	$EEEEEEEEEEE$

Heat shield computational results

Source	$\frac{PL}{A}\left[\frac{W}{cm}\right]$	Insulators
Hilal & Boom	68.6	$E_pE_pE_p$
Kokkolaras et al.	25.3	$NNNNNNNEEET$
Kokkolaras rerun	25.59	$NNNNNNTEETT$
Hilal & Eyssa	53.2	$E_pE_pE_p$
with stress constraint	24.55	$EEEEEEEEEEE$
with all constraints	23.77	$EEEEEEEEEEE$

Parameters:

- $\overline{T}_H = 300$ K, $\overline{T}_C = 4.2$ K, $L = 100$ cm, $n_{\max} = 10$
- $F = 250$ kN, $m_{\max} = 10$ kg, $\delta = 5\%$

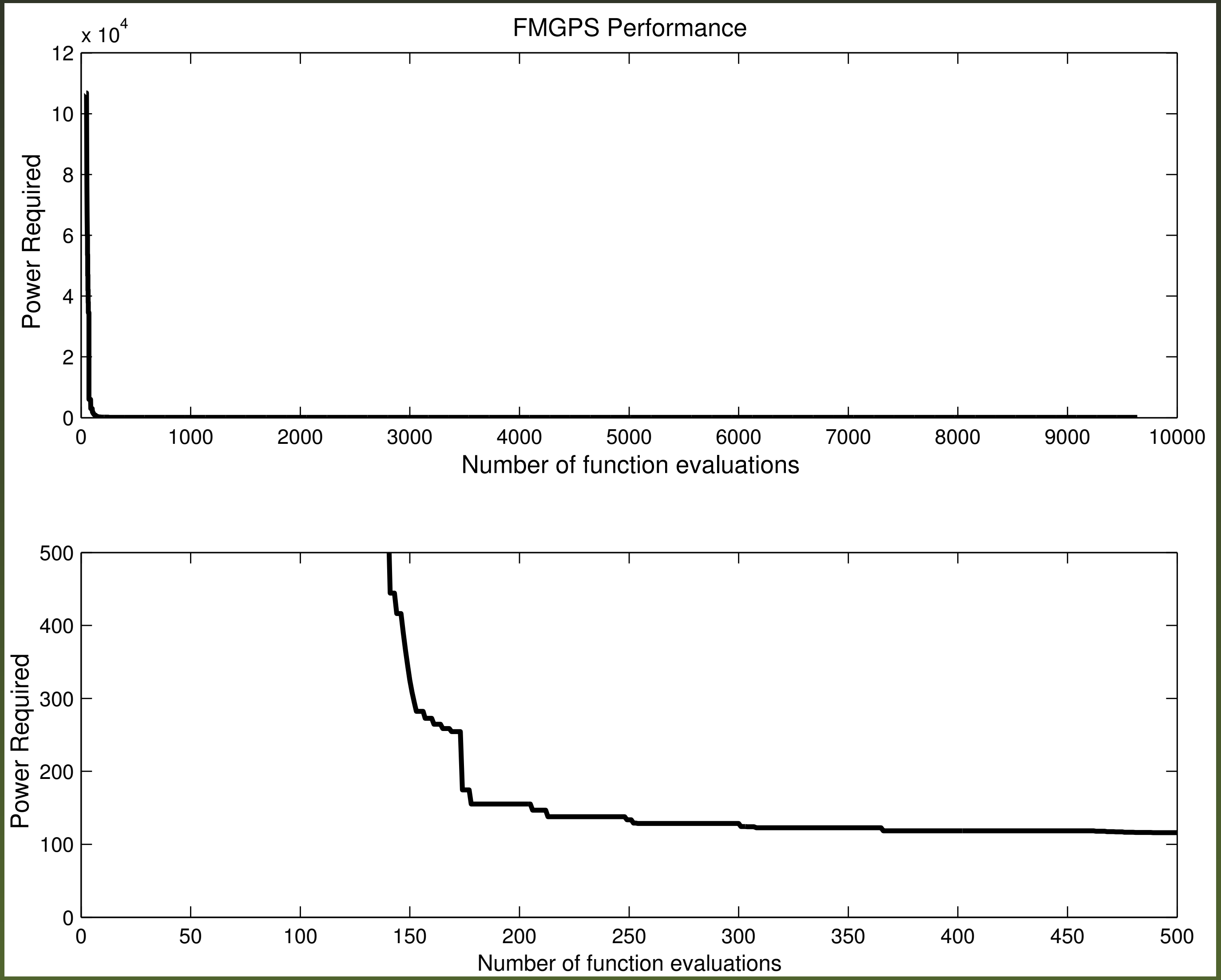
Heat shield computational results

Source	$\frac{PL}{A}\left[\frac{W}{cm}\right]$	Insulators
Hilal & Boom	68.6	$E_pE_pE_p$
Kokkolaras et al.	25.3	$NNNNNNNEEET$
Kokkolaras rerun	25.59	$NNNNNNTEETT$
Hilal & Eyssa	53.2	$E_pE_pE_p$
with stress constraint	24.55	$EEEEEEEEEEE$
with all constraints	23.77	$EEEEEEEEEEE$

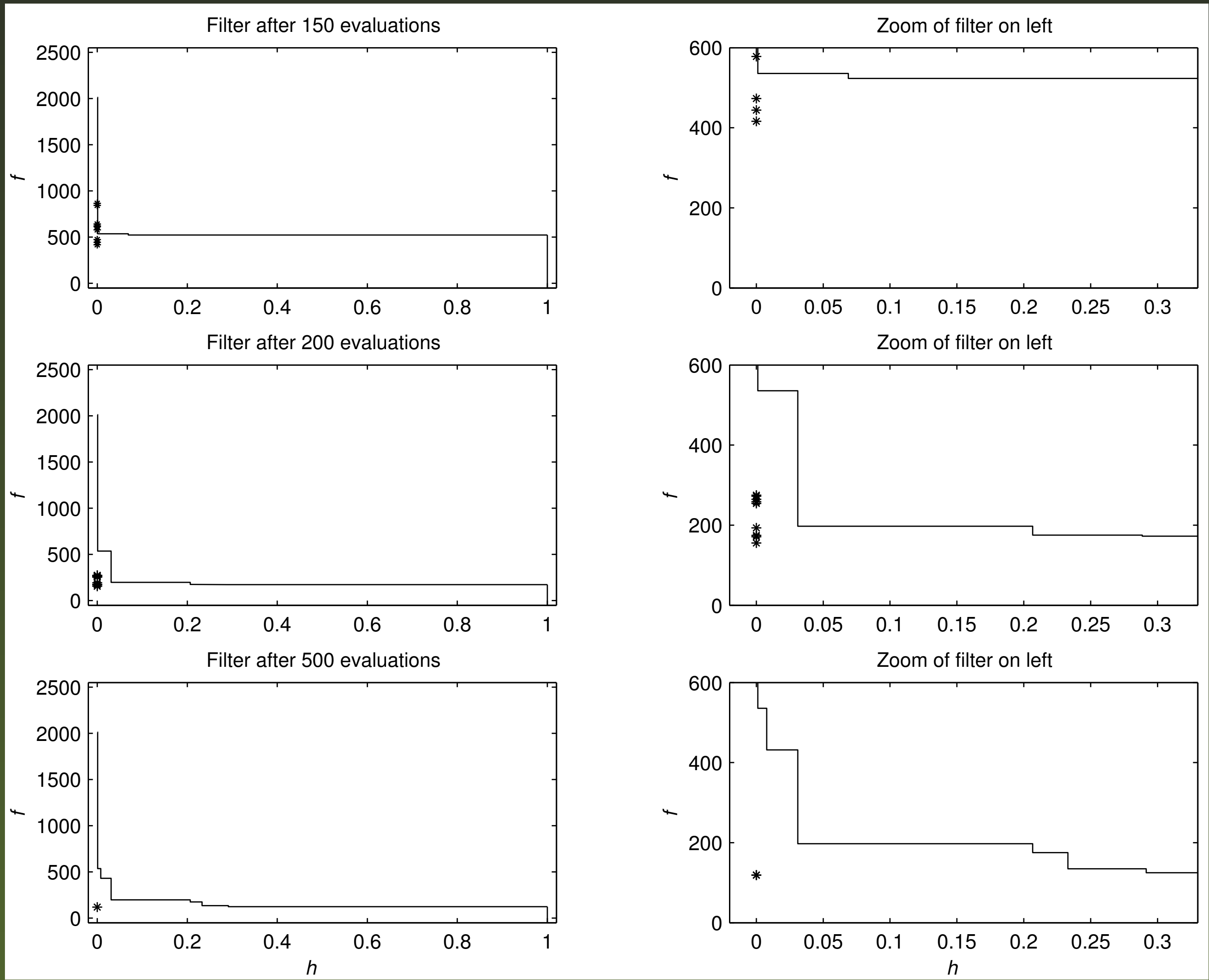
Parameters:

- $\overline{T}_H = 300$ K, $\overline{T}_C = 4.2$ K, $L = 100$ cm, $n_{\max} = 10$
- $F = 250$ kN, $m_{\max} = 10$ kg, $\delta = 5\%$
- Termination: $\Delta_k \leq .15625$

Profile of heat shield run



Heat shield filter progress



Conclusions

- We have found an effective method for mixed variable problems... BUT...

Conclusions

- We have found an effective method for mixed variable problems... BUT...
    - It requires serious thinking by the user as to what constitutes an acceptable solution for each problem

Conclusions

- We have found an effective method for mixed variable problems... BUT...
  - It requires serious thinking by the user as to what constitutes an acceptable solution for each problem
  - It is not easy to incorporate surrogates with categorical variables

Conclusions

- We have found an effective method for mixed variable problems... BUT...
  - It requires serious thinking by the user as to what constitutes an acceptable solution for each problem
  - It is not easy to incorporate surrogates with categorical variables
  - It is unlikely to be effective for many categorical variables

Conclusions

- We have found an effective method for mixed variable problems... BUT...
  - It requires serious thinking by the user as to what constitutes an acceptable solution for each problem
  - It is not easy to incorporate surrogates with categorical variables
  - It is unlikely to be effective for many categorical variables
- We have analyzed the method, BUT.., better results depend on the filter method for continuous variables

Conclusions

- We have found an effective method for mixed variable problems... BUT...
  - It requires serious thinking by the user as to what constitutes an acceptable solution for each problem
  - It is not easy to incorporate surrogates with categorical variables
  - It is unlikely to be effective for many categorical variables
- We have analyzed the method, BUT.., better results depend on the filter method for continuous variables